REMARKS

Claims 1-19 are pending in the present application, claim 18 having been added herein. The Office Action and cited references have been considered. Favorable reconsideration is respectfully requested.

The Office Action requires Applicant to submit a drawing to illustrate the claimed invention. Such a drawing is attached a new figure 1. The specification has been amended to specifically refer to that drawing.

Claims 16-17 were rejected under 35 U.S.C. § 101 as allegedly being directed to non-statutory subject matter. To advance prosecution, and without conceding the merits of this rejection, Applicant has amended claim 16 to depend from claim 1. Further, Applicant respectfully submits that claim 16 is tied to a machine, and is therefore statutory under the standard set forth in *In re Bilski*, 545 F.3d 943, 88 U.S.P.Q.2d 1385 (Fed. Cir. 2008). Specifically, claim 16 recites an application analysis server and a server for validation of applications. One of ordinary skill in the art would understand that the term "server" refers to a computer. Further, claim 16 recites "means for ensuring, ..., observance by this application of said validity criteria, an extraction of information being carried out on the application analysis server and an evaluation of said validity criteria being carried out on the server for validation of applications". Thus, this means has two components "an extraction of information" and "an evaluation of the validity criteria" that are "carried out" on the claimed servers, and therefore, the claim recites that the system includes functions that are performed by the computers claimed. The claim does not recite "software *per se*" – it does not claim an abstract idea, a mental process or

substantially all practical uses of a law of nature or a natural phenomenon. For at least these reasons, Applicant respectfully submits that claim 16, and therefor dependent claims 17 and 18, meet the requirements of 35 U.S.C. §101. Withdrawal of the rejection is respectfully requested.

Claims 1-18 were rejected under 35 U.S.C. §102(e) as being anticipated by Liang (U.S. Patent No. 7,120,572).

Claim 1 recites a method for determining the operational characteristics of a program, comprising a verification procedure comprising the following steps:

a first step comprising expressing the operational characteristics of the program as functions dealing with occurrences or sequences of occurrences of events occurring during executions of the program, these events being able to deal with particular operations, particular values of data, at particular program points and in particular states of the program, determining a level of precision with which these characteristics must be determined, determining a set of particular contexts of execution in which the program will always be executed, and determining operational specificities of a set of platforms on which the program will be executed,

a second step of estimation, by program analysis, and in consideration of the level of precision, of the set of particular contexts of execution and of the operational specificities of platforms, of information relating to the structure of the program, the execution paths of the program and to the values of data, at various points of the execution paths and under different execution conditions, of the states of the program and data handled by the program, and

a third step for determining the operational characteristics, by means of the information extracted by the program analysis, by computation of the functions

on the occurrences or particular sequences of occurrences of particular operations, dealing with particular values, at particular points of the program, in particular states of the program, for the set of execution paths determined by analysis. This is not taught, disclosed or made obvious by the prior art of record.

General remarks

Liang discloses "a program authoring system [that] preprocesses the program to verify the integrity of the program" by "determin[ing] whether any instruction in the program would violate the data type restrictions for that instructions" (Liang, abstract). It is an example of a bytecode verifier that Applicant lists as prior art that is unsatisfactory for Applicant's purpose (specification, [0025]). Liang's system, like other such systems, only verifies certain specific pre-established criteria that can be determined precisely.

Applicant's method, in contrast, allows for complex criteria expressed as "functions dealing with occurrences or sequences of occurrences of events", leading to an "estimation" to a variable "level of precision" (claim 1).

Claim 1

Liang discloses a "full program verifier for verifying whether or not a specified program satisfies certain predefined integrity criteria" (col. 5, ll. 9–11). These criteria are "data type and stack usage restrictions of the language in which the program is encoded" (col. 10, ll. 11–14). For the Java (bytecode) language which is the focus of Liang's disclosure, said restrictions are "operand data type compatibility and proper stack manipulations" (col. 6, ll. 58–62). The verifier detects operand stack overflow and underflow conditions as well as operand data type

checking so that the program interpreter need not performs corresponding checks during program execution (col. 10, ll. 38–47). Said operand data type checking consists of a comparison between the data type requirements of instructions and the data types of said instructions' operands as determined by an analysis (col. 15, ll. 10–19).

Applicant's method comprises a first step of expressing the operational characteristics of the program as functions dealing with occurrences or sequences of occurrences of events. An example of such characteristic is that any call to a certain specific method, say “commitTransaction()”, must be preceded by a call to another specific method, here “beginTransaction()” (specification, [0051]); this characteristic cannot be expressed in terms of Liang's disclosure. Furthermore, Applicant's method allows for the expression of arbitrarily complex criteria (specification, [0049]) that the author of the verification method may not have predicted, for example through the use of logic formalism such as linear temporal logic (specification, [0050]). Thus Liang does not teach expressing the operational characteristics of the program as functions dealing with occurrences or sequences of occurrences of events.

Liang discloses a “full program verifier for verifying whether or not a specified program satisfies certain predefined integrity criteria” (col. 5, ll. 9–11). These criteria are “data type and stack usage restrictions of the language in which the program is encoded” (col. 10, ll. 11–14). These criteria are binary, that is, either they are satisfied or they are violated: for example, either a program exhibits a stack underflow, or it does not (col. 15, ll. 10–19). Applicant's method involves more complex characteristics that need not be completely determined; for example, if a characteristic is that a certain amount must be positive, it is sufficient to determine

that said amount is between 5 and 20, without enumerating all possible values of said amount (substitutes specification, p. 17, ll. 5-11).

Applicant's method therefore includes a step of determining a level of precision with which these characteristics must be determined, which is not taught by Liang.

Liang discloses "a program authoring system [that], prior to distributing a program, preprocesses the program to verify the integrity of the program" (abstract). Said program authoring system operates independently of the client system where the program will be run. Liang notes that a wide variety of client systems is possible (col. 3, ll. 43–54; col. 6, ll. 45–48). Using the Java language, a single version of the program is distributed to many different clients (col. 6, ll. 55–57).

In Applicant's terminology, Liang's program authoring system operates in a manner independent of any particular context of execution in which the program will be executed. For these reasons, Liang does not teach determining a set of particular contexts of execution in which the program will always be executed. Furthermore, Liang discloses that Java language instructions must only be executed when the operand stack contains values in sufficient number and of appropriate data types (col. 6, ll. 18–32): otherwise the program would fail during execution (col. 7, ll. 10–20). It will be noted that the constraints on the use of Java instructions are an inherent property of said instructions, for example an addition instruction always requires two numbers (col. 5, ll. 49–57). Thus Liang does not teach determining a set of particular contexts of execution for instructions, much less for programs.

Liang discloses that some programs are "platform specific", that is, "programs executed on one [platform] will not be executable on the others" (col. 6,

ll. 33–48). This is however generally not the case for programs using the Java language (col. 6, ll. 55–57) which underlies most of Liang's disclosure. Liang does not teach how to detect or handle platform specific program behavior when it does arise.

Thus Liang does not teach determining operational specificities o a set of platforms on which the program will be executed.

As discussed above, Liang discloses a "full program verifier for verifying whether or not a specified program satisfies certain predefined integrity criteria" (col. 5, ll. 9–11), said criteria being binary, such as whether data type restrictions are violated (col. 27, ll. 33–40), whereas Applicant's method allows for complex characteristics that need not being completely determined.

Thus Liang does not teach estimation (…) in consideration of said level of precision. As discussed above, Liang teaches a verifier that operates in a manner independent of any particular context of execution in which the program will be executed and possible operational specificity of the platform on which the program will be executed (col. 6, ll. 55–57). Thus Liang does not teach estimation (…) in consideration of (…) said particular contexts of execution and of said operational specificities of platforms.

As discussed above, Liang does not disclose operational characteristics such as expressed in the first step of Applicant's claimed method; in particular Liang does not teach determining said operational characteristics (…) by computation of said functions on the occurrences or particular sequences of occurrences of particular operations.

Then for at least one of these reasons, claims 1 is not anticipated by Liang.

Claims 2-15 being dependant of claim 1, for the same reasons, claims 2 - 15 are not anticipated by Liang.

Claim 16 claims a system to implement all the steps of the method of claim 1, then for the same reasons, claim 16 is not anticipated by Liang.

Claims 17 and 18 being dependant of claim 16, for the same reasons, claims 17 and 18 are not anticipated by Liang.

For at least these reasons, Applicant respectfully submits that claims 1, 16, and 19 are patentable over the prior art of record whether taken alone or in combination as proposed in the Office Action. Claims 2-15 and 17-18 are believed to be patentable in and of themselves, and for the reasons discussed above with respect to claims 1 and 16, from which they depend.

In view of the above amendment and remarks, Applicant respectfully requests reconsideration and withdrawal of the outstanding rejections of record. Applicant submits that the application is in condition for allowance and early notice to this effect is most earnestly solicited.

If the Examiner has any questions, he is invited to contact the undersigned at 202-628-5197.

Respectfully submitted,

BROWDY AND NEIMARK, P.L.L.C.
Attorneys for Applicant(s)

By /Ronni S. Jillions/
Ronni S. Jillions
Registration No. 31,979

RSJ:ma
Telephone No.: (202) 628-5197
Facsimile No.: (202) 737-3528
G:\BN\M\Mout\Vetillard2\Pto\2009-09-09Amendment.doc